# Contents

## Introduction
- How This Guide Is Organized ...................................................... 5
- Related Documentation ............................................................... 6

## How the Components Fit Together
- A Typical Marimba Infrastructure .................................................. 7

## Designing Your Marimba Infrastructure
- Determining the Requirements for the Infrastructure Setup ..................... 13
- Identifying Business Objectives ....................................................... 13
- Adding Infrastructure Components to Your Architecture Diagram ............... 14
- Anticipated Infrastructure Platforms and Hardware ............................ 20

## Performance Considerations
- Deciding Which to Use: Repeaters, Mirrors, or Proxies .......................... 23
- What the Replication Mechanisms Have in Common ........................... 24
- Repeaters: Pros and Cons .......................................................... 24
- Mirrors: Pros and Cons .............................................................. 25
- Marimba Proxies: Pros and Cons .................................................. 26
- Deployment Examples .................................................................... 26
- Determining Whether Your System Can Support the Logging Feature ........... 31
The purpose of this guide is to help you design a Marimba infrastructure for your enterprise. Depending on the size and complexity of your environment, you might not require the assistance of Marimba Professional Services to set up your infrastructure. In this case, you can use this guide to determine which Marimba components to place at various company sites, and then you can use the Deployment Guide to actually deploy and install those components. (Even if you will be getting assistance from Marimba Professional Services, you can still use this guide to better understand the issues involved.)

**Note:** If you have more than 3,000 endpoints that you want to manage by using Marimba products, it is recommended that you work with Marimba Professional Services to create an architecture plan, rather than relying solely on this document. In this case, you can contact Marimba Technical Support to arrange for a customer environment assessment.

### How This Guide Is Organized

The first chapter, *How the Components Fit Together* on page 7, presents the typical infrastructure setup for a small or medium-sized enterprise. All the Marimba components are defined, and a diagram shows how the components interact.

The second chapter, *Designing Your Marimba Infrastructure* on page 13, helps you determine whether the typical infrastructure setup will work for your enterprise. This chapter helps you assess the performance load on various...
components and discusses the issues involved. Formulas are provided to help you make decisions regarding the hardware and software you plan to dedicate to the Marimba infrastructure.

The last chapter, Performance Considerations on page 23, discusses the various performance issues that you need to address when setting up the infrastructure. Some alternative configurations are presented.

Once you finish working your way through this guide, you should have a good idea of which machines you will use for the various components and whether you need to purchase additional hardware and software. You will have a plan that you can then implement by following the instructions in the Deployment Guide, available on Marimba’s Documentation page (www.marimba.com/doc/).

Related Documentation

In addition to the Deployment Guide, mentioned earlier, you will need to use the System Requirements guide to make sure that the machines you are planning to use in your Marimba infrastructure satisfy the minimum hardware and software requirements. You can access both of these documents from Marimba’s Documentation page (www.marimba.com/doc/).
This chapter describes the standard topology that Marimba recommends for a typical enterprise, and defines the hardware and software components that most customers require. If, when reviewing the diagram presented here, you realize that your enterprise requires a different strategy, it is recommended that you contact Marimba Technical Support. You might need the assistance of Marimba’s Professional Services to design an architecture that meets your requirements.

A Typical Marimba Infrastructure

The diagram shown in Figure 1 on page 9 depicts the architecture that is typically recommended for deploying a Marimba infrastructure. In order to understand the diagram, read the following definitions, which will help you become familiar with the components involved.

**Console server.** Administrators use the Marimba console to manage changes to software, content, and configuration on endpoints, as well as to collect inventory information. The browser-based user interface lets you access Marimba’s infrastructure management tools, including Report Center and Subscription Policy Manager. In this document, the term *console server* means the machine on which the Marimba console is installed. For Server Management customers, Deployment Manager is also commonly installed on the console server. The Marimba console is the primary place where user interaction with Marimba products occurs. You use the console to configure almost all the other components.
Master distribution server (transmitter). The master distribution server (commonly known as the master transmitter in a Marimba infrastructure) functions similarly to a Web server, except that instead of serving Web pages, it serves applications and content that have been packaged into “channels.” The channel format enables cross-platform support and byte-level updates to software and content. Because the transmitter hosts all the content and applications that you distribute to endpoints, it is the heart of your Marimba infrastructure.

Mirrors. Mirror transmitters can be set up to regularly auto-replicate all their content from the master transmitter (or master/mirror farm). Mirrors provide high availability and scalability. (Mirrors and repeaters are actually regular transmitters what have undergone some simple configuration changes that allow them to function in a new capacity.)

Repeaters. Repeater transmitters come into play when you set up a master to redirect client requests to a repeater. Repeaters regularly auto-replicate some or all of the content from a master/mirror farm. The client redirection provides the ability for clients to communicate with the nearest server, and it provides built-in fault tolerance.

Proxies. Marimba proxies are a good choice in remote locations where endpoints do infrequent updates or do not require a majority of corporate channels (either initially or on an ongoing basis). You might also want to use a Marimba proxy in a remote location where hardware resources are limited. In this situation, you can host the proxy on a desktop-class machine to service up to 50 endpoints, and you can perhaps also use the machine to run additional applications and services.

Endpoints. The term endpoints in this document refers to both desktop endpoints (desktops and laptops) and server endpoints. You will place a Marimba agent (commonly known as a tuner) on every endpoint. It is through this agent that targeting, installations, and updates are carried out.

After you study the diagram on the next page, you can proceed to the next chapter, which walks you through the process creating your own architecture diagram and deciding where to place Marimba infrastructure components in your enterprise.
A Typical Marimba Infrastructure

Figure 1. Recommended System Architecture for Standard Environments

A description of the process depicted in this diagram appears on the next page. The numbers in the diagram correspond to the step numbers in the text.
1 After developing and testing channels (packaged applications and content), you copy the channels from your distribution server (transmitter) in the testing lab to the master production transmitter.

Copying channels from the test or QA lab to your production environment is a task you will do on an ongoing basis, usually weekly or monthly, or whenever new content is available.

2 You use the Marimba console to publish the following plug-ins to the master transmitter:

a. You use Report Center (part of the Marimba console) to publish the Inventory and Logging plug-ins (which contain schedules for inventory scanning and log collection) to the master transmitter.

You might publish these plug-ins from time to time if you need to change a configuration setting, such as the schedule for an inventory scan or log collection from endpoints.

b. You use Subscription Policy Manager (part of the Marimba console) to publish the Subscription plug-in to the master transmitter.

3 Channels (and their plug-ins) are then automatically replicated to the mirror transmitters (according to a schedule).

In the diagram, two of the mirrors are placed, along with the master transmitter, behind a load balancer. A third mirror is placed at a different site for disaster recovery purposes, so that it can quickly be promoted to the master if the master goes down or there is a problem at the main data center.

Alternative: If you do not want to use a load balancer to provide fault tolerance, one workaround is to have only one master that replicates to repeaters and then have a mirror that is not part of the rotation. In this case, the mirror is not really used by endpoints unless the master goes down. Then you would quickly convert the mirror to a master by using Transmitter Administrator, a fairly simple operation.

4 Channels are then replicated to the repeaters at branch offices (according to a schedule). Channels are cached on proxies at remote sites (on an on-demand basis). Plug-ins are replicated to repeaters, but not to proxies.

In this scenario, the master and mirrors are placed behind a load balancer. The load balancer is identified by a single IP address. When repeaters connect to that IP address for replication, they are automatically routed to the master or one of the mirrors.

Note: If you are a Server Management customer, you will probably not use repeaters because you will want to control exactly which distribution server endpoints use for content installation and updates.
The endpoints download new channels and update current ones:

- If the endpoints use subscription policy management, they get the channels according to their subscription policies.
- If the endpoints are part of a Server Management system, then rather than using the subscription policies, the endpoints get their channels according to the job schedule set by Deployment Manager.

Inventory scan reports and log messages are sent back from the endpoints to the plug-ins on the repeaters. The repeaters then forward the data to the plug-ins on the master and mirrors. (Endpoints that use proxies send their scan reports and logging data back to the master and mirrors, since proxies cannot perform plug-in tasks.)

The master and mirror plug-ins then insert the data collected from endpoints into the database.

Report Center (part of the Marimba console) retrieves data from the database in one or more of the following ways:

- You create reports for hardware and software inventories, software usage (if you have the Software Usage component), policy compliance, and so on. You can create your own reports or use predefined reports from the Query Library.
- Previously created reports are automatically run according to schedules, and the results are e-mailed to the appropriate people.
- You build queries that return a group of machines, and you save these queries in special predefined folders. These queries are then used by Marimba’s other management tools, such as Deployment Manager, Transmitter Administrator, Tuner Administrator, to allow you to manage multiple machines simultaneously.

Report Center runs scheduled queries to create collections in a directory service. These collections are then targeted by the Subscription module.

The Marimba console retrieves user information from the directory service (to authenticate users).

Because you can use the Marimba console to assign various roles to users, administrators will have access to only certain portions of the management console and directory service schema tree, corresponding to each administrator’s duties and responsibilities.

The Subscription plug-in resolves group membership and retrieves subscription policies from your directory service.
A Typical Marimba Infrastructure
Reviewing the typical Marimba infrastructure setup described in How the Components Fit Together on page 7 gives you some idea of the strategies recommended for addressing issues such as disaster recovery and performance bottlenecks in an enterprise infrastructure. That chapter introduced you to a basic infrastructure architecture plan. This chapter helps you determine whether that plan will work for you. If the plan needs modifications to work in your enterprise, you might need to contact Marimba Technical Support, and arrange for assistance from Marimba Professional Services.

**Determining the Requirements for the Infrastructure Setup**

Even within the basic plan (described in How the Components Fit Together on page 7), there are choices to be made, such as how many distribution servers and proxies to use and where to put them. This chapter helps you make those decisions.

**Identifying Business Objectives**

Writing down the basic business requirements for your Marimba infrastructure will help you decide how much hardware and software to devote to the project. For example, if your business objective is to update 100 GB of Web content on 500 servers every day, your hardware requirements might be greater than if your objective is to update a 100 MB custom application on 2000 desktops once a month.
Determining the Requirements for the Infrastructure Setup

Your business objectives might include one or more of the following:

- Maintain accurate asset inventories of all hardware and software.
- Distribute OS patches to UNIX servers.
- Deploy and manage applications on remote desktops and laptops. These applications consist of both custom applications and shrink-wrapped applications, such as Microsoft Office.
- Streamline the current software distribution process.
- Reduce support costs.

At the time when you purchased Marimba products, you discussed your requirements with a sales representative, and that person helped you determine which products to buy. At this point, you will review your requirements in order to determine where and how to use the products you purchased.

Adding Infrastructure Components to Your Architecture Diagram

When creating an architecture plan for your Marimba infrastructure, you will first use some basic formulas for estimating how many of each component you need, as described in Machine Count and Location Requirements on page 14. You can then consider some issues that might affect those original estimates, as described in Estimated Service Load on page 17 and Security Requirements on page 19. When your plan is finalized, the last step is to make sure that your existing hardware and software meet the minimum system requirements, as described in Anticipated Infrastructure Platforms and Hardware on page 20.

Machine Count and Location Requirements

At this point, you need to figure out whether the setup shown in Figure 1. Recommended System Architecture for Standard Environments on page 9 will work for your enterprise. That diagram shows a number of master, mirror, and repeater distribution servers at various regional and branch offices. The number of distribution servers and proxies you need depends on the number of endpoints you have and the number of remote sites in your enterprise.

Therefore, create a system diagram similar to the one in Figure 2 on page 15. The diagram needs to show the following things:

- Corporate headquarters and the various regional and branch sites
- The speed of the network connections between the sites
- Location of the database and directory service
Designing Your Marimba Infrastructure

Determining the Requirements for the Infrastructure Setup

- Which sites replicate the directory service
- How many endpoints are located at each site

After creating a diagram of your enterprise and recording the number of endpoints at each site, you can use those numbers in formulas to help you figure out how many mirrors, repeaters, and proxies you need, as described in the following paragraphs.

Requirements for the master/mirror farm. The process shown in Figure 1. Recommended System Architecture for Standard Environments on page 9 begins with the distribution servers (transmitters). To help you determine how many distribution servers you need, use this formula: For every 7,500 endpoints in your enterprise, you need one master or mirror behind the load balancer.

According to this formula, how many machines do you need for your master/mirror farm? (The minimum number is 2.)

Note: This formula of estimating that you need one master or mirror for every 7,500 endpoints is a way of making a rough estimate only. The actual number of endpoints that a master can accommodate varies considerably, depending on the update frequency and the size of the updates. A master might be able to handle 50,000 endpoints if the updates were infrequent and the changes were small.
Determining the Requirements for the Infrastructure Setup

Alternative: If the total number of endpoints is under 5,000 and you do not have or do not want to use a load balancer, you can alternatively use one master and then have one mirror as a backup/“hot spare” distribution server. The mirror provides fault tolerance by maintaining workspace backups of the master and can quickly be promoted to master in the event that the master goes down. This strategy requires, however, a manual DNS alias change when promoting the mirror to a master.

Requirements for the console server. Figure 1 on page 9 shows only one console server. In this typical setup, you would have one Report Center (part of the Marimba console), and all administrators would have the same inventory queries available to them when they log in. When they run queries, the reports might contain results from endpoints in one or more domains, depending on how the query is constructed.

If, however, your enterprise uses Active Directory and supports a multi-domain forest, and you want to restrict administrators in one domain from seeing queries that are to be used only by administrators in another domain, you can choose to install multiple console servers (and Report Centers). The setup described in this guide assumes you will have only one console server. If you want to use more than one, you will need more assistance than is available in this guide.

Requirements for repeaters. For each remote site, plan to use one or two repeaters. Although a repeater has the same capacity as a master (and can handle 7,500 endpoints), for disaster recovery purposes, you might want to have a second repeater at a site. According to this formula, how many machines do you need for hosting repeaters?

Requirements for proxies. Consider using a proxy at remote locations, under one of the following conditions:

- Endpoints do infrequent updates or do not require a majority of corporate channels (either initially or on an ongoing basis).
- Hardware resources are limited. In this situation, you can host the proxy on a desktop-class machine to service up to 50 endpoints, and you can perhaps also use the machine to run additional applications and services.
- The site has a slow link (such as a dial-up modem) to the main corporate data center.

Taking these considerations into account, how many machines do you need for hosting proxies?

Now that you have worked through these questions, read the next section to find out whether you need to make any adjustments to your plan.
Estimated Service Load

Load on the Subscription plug-in and master/mirrors. To determine whether your current estimate for master, mirrors, and repeaters is sufficient, answer these questions:

- How often will updates and new distributions need to be sent to endpoints—daily? weekly? hourly? Is there a time window during which distributions can be made, or can they be made anytime?

  The answers to these questions affect how often subscription runs and contacts the Subscription plug-in on the distribution servers. If you determine that, for example, the Subscription plug-in will be getting 10,000 requests every 90 minutes (90 minutes is the default subscription interval), you might need to change your plan. For example, in this scenario, say you were planning to have all 10,000 endpoints contact the Subscription plug-in on the master, then there might be a problem. To solve it, you might increase the subscription interval, so that endpoints contact the plug-in every 180 or 360 minutes. Or you might decide to have the repeaters replicate the Subscription plug-in. This way the 10,000 endpoints would not all contact one plug-in.

- How many queries will be made to the directory service per day? Use the following formula for each instance of the directory service:

  \[(\text{Number of times per day subscription runs}) \times (\text{number of endpoints})\]

  Desktop and laptop endpoints will most likely get their new software distributions and updates by using a subscription policy. In order for endpoints to get the plan (and their updates and new software), the endpoints need to run the Subscription Policy Service at regularly scheduled times. How often this service runs depends on how quickly you want software to be deployed once it is ready. The default is to run the service every 90 minutes. Therefore, if you use this default for 1,000 endpoints, the directory service will get 1,000 requests every 90 minutes.

  When you use the formula, if the resulting number is greater than 10,000 every 90 minutes, increase the subscription intervals to 180 or 360 minutes.

- How many megabytes (estimated range) will be sent for each distribution?

  This question is more important for Server Management systems, which might replicate very large files and large numbers of files in a single channel. To find out the limits regarding channel size and number of channels a distribution server can host, see the Server Management release notes, available on Marimba’s Documentation page (www.marimba.com/doc/).
If your master will host very large channels or large numbers of channels, you will need to make sure that the machine hosting the master has adequate processing power and space.

**Load on the console server and directory service.** To determine whether your current plans for the console server and directory service are sufficient, answer these questions:

- **How often will collections be scheduled to run?**
  Report Center runs collections according to a schedule and then saves the results in your directory service. (The results are used as target groups by Subscription Policy Manager when sending subscription policies to endpoints.) Therefore running collections involves the console server, the directory service, and the database. It is recommended that collections be run no more often than once a day, during off-peak hours so that the directory service will not be overburdened.

- **Will repeaters be able to query a locally replicated directory service, to limit WAN traffic and provide faster service to endpoints?**
  If not, you might need to configure repeaters so that they do not replicate the Subscription plug-in. That way, all endpoints will contact a plug-in from the master/mirror farm, and thereby query only the main directory service.

**Load on the Inventory plug-in and master/mirrors.** To determine whether your current plan for the master/mirror farm is sufficient, answer this question:

- **How often will inventory scans be done?**
  It is recommended that scans be run once a day. Each time an inventory scan is done on an endpoint, a scan report is sent to the Inventory plug-in on the repeater, and then the scan is usually forwarded to the master/mirror farm. If you anticipate having more than 10,000 scan reports sent at the same time (that is, using the same schedule), consider one of the following options:

  - **Schedule different components to be scanned according to different schedules.** For example, you can set the system/hardware components to be scanned daily, but set the applications to be scanned only once a week.
  
  - **Set repeaters to insert scan reports directly into the database, rather than forwarding them to the master.** This option is not recommended, however, if the repeater would need to go across a WAN to insert data in the database. When a repeater inserts data directly, it uses SQLNet to communicate with the database, which uses more bandwidth than Marimba’s protocol. Marimba’s protocol is used when forwarding data to a master.
Note: The transmitter or transmitters that insert data in the database should be on the same subnet as the database server. This is not required, but it greatly improves performance.

**Load on the database.** To determine whether you need to have a database dedicated to your Marimba infrastructure, make the following calculations:

- Required space for initial inventory scans is: ____
  
  Use this formula: \((\text{Number of endpoints}) \times (100 \text{ KB})\)

- Required space for subsequent scans (differential scans) is: ____
  
  Use this formula: \((\text{Number of endpoints}) \times (20 \text{ KB})\)

- Total required database storage for inventory scans is: ____
  
  Use this formula: \(\text{Initial scan space} + \text{diff scan space}\)

  A minimum of 3 GB is recommended.

Will the database be able to accommodate centralized logging data?

**Tip:** To answer this question, see Determining Whether Your System Can Support the Logging Feature on page 31.

Depending on your answers, you might need to increase the capacity of your database server, or, conversely, you might discover that you do not need to dedicate the database to the Marimba infrastructure. If the volume of centralized logging data is an issue, you might be able to adjust the level of log collection so as not to overload your database.

**Security Requirements**

Firewall issues can affect which components you install and where you install them. For example, your environment might require you to place a reverse proxy outside the firewall, rather than allowing endpoints to contact a distribution server directly through the firewall. In general, the following of ports must be open:

- Listener port (default is 5282). This port needs to be open so that the master/mirror farm can be contacted for new channels and updates.

- (For Server Management systems) Deployment Manager status port (default is 8000). This port needs to be open so that endpoints can send status messages back to Deployment Manager.

- (Rarely) Administration port (default is 7717). This port needs to be open only if you are trying to manage a component or publish a channel from outside the firewall.
If having these ports open is an issue for your environment, you need to contact Marimba Customer Support, to discuss alternatives.

For more information about reverse proxies, see Reverse Proxy Outside a Firewall on page 29.

Additional Suggestions

By this time, you should have gained some understanding of the various server components in a Marimba infrastructure: masters, mirrors, repeaters, and proxies. If, however, you would like more details about these components, in order to help you decide which ones to use in which locations, see Deciding Which to Use: Repeaters, Mirrors, or Proxies on page 23.

Once you have determined which types of components to use, the final step is to determine whether your current hardware and software will support these components, as described in the next section.

Anticipated Infrastructure Platforms and Hardware

As you worked through Machine Count and Location Requirements on page 14, you created an architecture diagram of your enterprise. At this point, you should add to that diagram the Marimba components that you want to place at the various company sites. After the diagram is complete, determine whether you already have the required hardware or will need to purchase new hardware or upgrade existing machines.

The procedure that follows will help you. It asks you to find out whether your existing machines satisfy the minimum system requirements for the type of component you want to install. These minimum requirements are listed in the System Requirements document, available on Marimba’s Documentation page (www.marimba.com/doc/).

To determine your hardware and software requirements:

1 Master/mirror farm. Make a list of the machines that you want to use for the master and mirrors. To determine whether the machines in your list satisfy the system requirements for Marimba distribution servers (transmitters), refer to the System Requirements document, available on Marimba’s Documentation page (www.marimba.com/doc/).

2 Test/QA distribution server. Make a list of the machine or machines that you want to use for hosting channels during testing and QA. Do the machines that you want to use as test transmitters satisfy the system requirements?
3 **Repeaters.** Make a list of the machines that you want to use for repeaters. Do these machines satisfy the system requirements?

4 **Proxies.** If you plan to use Marimba proxies, make a list of machines that will host proxies. Do these machines satisfy the system requirements?

5 **Console server.** Does the machine that you want to use for hosting the Marimba console satisfy the system requirements?

6 **Endpoints.** Make a list of the various operating systems that your endpoints run on. Are all of these operating systems supported for a Marimba infrastructure?

   If not, you need to either upgrade the endpoint to a supported version or find out whether some other version of Marimba products supports that operating system.

7 **Packaging machine for creating software packages.** Packaging machines are used to convert existing applications to Marimba channels. When you package software, the packaging machine needs to use the same operating system as the endpoints that will be receiving the software packages.

8 **Packaging machine for creating content packages** (for Server Management). The system requirements of this machine depend in part on how much content (data files) you want to convert into Marimba data channels. Note that this machine is usually just one of your Server Management endpoints, and so the system requirements are the same.

Once you have designed your Marimba infrastructure, you will be ready to install the components and deploy distribution agents to your endpoints. For instructions regarding these tasks, see the *Deployment Guide*, available on Marimba’s Documentation page (www.marimba.com/doc/).

**Note:** If you have more than 3,000 endpoints that you want to manage by using Marimba products, it is recommended that you work with Marimba Professional Services to create an architecture plan, rather than relying solely on this document. In this case, you can contact Marimba Technical Support to arrange for a customer environment assessment.

Even if you have fewer than 3,000 endpoints and are able to create your own infrastructure plan, you might want to have Marimba Professional Services validate your design.
This chapter delves more deeply into some of the issues raised in previous chapters. (In those chapters, cross-references were made to the relevant sections of this chapter.) This chapter also discusses some alternative architecture choices. However, it is not required reading. If you have already successfully made an infrastructure architecture diagram, you might not need to read this chapter. It contains the following sections:

- **Deciding Which to Use: Repeaters, Mirrors, or Proxies** on page 23. Although earlier chapters provided recommendations regarding these components, you might have some special circumstances that were not addressed. This section provides several architecture examples for additional situations.

- **Determining Whether Your System Can Support the Logging Feature** on page 31. In a Marimba infrastructure, log messages generated on endpoints can be regularly collected in a central database. This feature is very powerful but needs to be used correctly so that your database is not filled up too quickly. This section steps you through the issues.

## Deciding Which to Use: Repeaters, Mirrors, or Proxies

The purpose of this section is to help you use replication effectively in your environment, whether to achieve load balancing, to bring channel content geographically closer to your end users, or to protect sensitive information behind a firewall. Replication is the transfer of channels from a master to a mirror or repeater. For the purposes of this discussion, “replication” could also be said to include the caching of channels on a proxy. Because you will need to
choose between replication mechanisms, or choose a combination of them, this section first gives an overview of their differences, and then gives examples of various architecture solutions.

**What the Replication Mechanisms Have in Common**

In order to appreciate the differences between repeaters, mirrors, and proxies, it is important to first understand the capabilities they have in common:

- They can all retrieve content from a master distribution server (transmitter).
- They can also cache data geographically closer to endpoints, so that, for example, no endpoint has to leave the LAN for data.
- For all three, you always publish updates to only one location—the master. Repeaters and mirrors automatically replicate content from the master. Proxies cache content from the master when an endpoint requests a channel.

**Repeaters: Pros and Cons**

The way repeaters work is that when an endpoint sends a request to the master, that transmitter doesn’t usually service the request itself but rather sends a list of repeaters back to the endpoint. The endpoint then contacts the first repeater on the redirection list. If that repeater is unavailable for some reason, the client contacts the next repeater on the list. The advantages of this strategy are:

- Because a group of repeaters is available, if one or more become unavailable, the endpoint can still get the request serviced by another repeater on the list.
- The entire redirection process is completely transparent to endpoints. That is, if you add or remove repeaters, you won’t need to change any settings on the endpoints.

The disadvantage is:

- Because the repeater system relies on the fact that each repeater has a separate IP address, repeaters do not function well with third-party redirection mechanisms (load balancers).

The list of repeaters that the master sends to endpoints contains either a DNS name or an IP address for each repeater. The problem with this strategy is that third-party redirectors that perform DNS round-robin can
associate one DNS name with any one of several revolving IP addresses. An equally problematic situation arises with another type of third-party redirector, which can hide a pool of servers under one IP address.

**Note:** Although repeaters are usually used to replicate channels from a master, you can also publish a channel directly to a repeater. In this case, the channel you publish to the repeater stays local to that repeater; it does not get replicated back to the master or to other repeaters.

### Mirrors: Pros and Cons

Mirrors are exactly like regular (master) transmitters, except with respect to the way that they get their content. At pre-configured intervals, a mirror copies all the channels from a master. You can also configure a mirror to replicate the publish and subscribe permissions of the master.

The advantage is:

- Mirrors can be used by third-party load-balancing products that support such strategies as DNS round-robin and by routing products that hide a pool of servers under a single IP address. This is the recommended strategy—to place one or two mirrors and a master behind a load balancer.

The disadvantages are:

- When a new mirror is added, existing endpoints cannot use it until they explicitly switch to that transmitter. There is no automatic redirection (unless a third-party redirector is used).

- A mirror can mirror channels from only one master, whereas repeaters can replicate channels from many masters.

- Publishing is disabled for mirrors. That is, you cannot use a mirror to simultaneously mirror a master and host other channels that are not on the master. Publishing is *not* disabled for repeaters.
Deciding Which to Use: Repeaters, Mirrors, or Proxies

Marimba Proxies: Pros and Cons

The way proxies work is that when an endpoint requests a channel, the proxy gets the channel from the master, saves a copy in its cache, and then delivers the channel to the endpoint. The next time an endpoint requests the channel, the proxy sends the copy that is in its cache, if it's still current. If the copy in the cache is not current, the proxy updates it and then sends the updated copy to the endpoint. The advantages are:

- You can use proxies to cache content from an unspecified number of transmitters, whereas repeaters can only work with channels that have been specified as repeatable. Mirrors only mirror the channels from one master.
- Proxies require less administration time than mirrors or repeaters. Normal (forward) proxies can handle all channel update requests from any transmitter.
- Proxies use bandwidth more efficiently than repeaters and mirrors because proxies only request files from the master when an endpoint requests an update. Therefore, if channels are republished frequently (for example, once a day) but endpoints check for updates only occasionally (for example, once a week), the master would be contacted for updates less frequently by a proxy than by a repeater.

The disadvantages are:

- Requests for files must first go to the master (in order to find out whether the proxy has the latest version of the files).
- Proxies don’t perform plug-in processing. All plug-in processing needs to be done by the master, whereas repeaters and mirror transmitters can run plug-ins.
- Proxies require settings to be explicitly made on the endpoints that use them, whereas repeaters can be added or removed without requiring any changes to be made on endpoints.

Deployment Examples

The following four examples show how you might use some combination of proxies, reverse proxies, repeaters, and mirrors to satisfy common infrastructure requirements.
**Example 1.** You have some channels that use CPU-intensive plug-ins, and you want to deploy these channels over an intranet at a single site. You also want to provide automatic redirection in case one or more transmitters become unavailable.

**Example 1. Round Robin Redirection Strategy**

In this situation, you would use round-robin repeaters or mirrors with third-party redirectors that can perform DNS round-robin. Proxies would not be able to distribute the plug-in load.
**Example 2.** You have a remote office on another continent and you want to reduce the overseas bandwidth between it and the central site. The channels are not updated frequently.

![Diagram of network architecture with Master (in London), One or More Proxies (in Tokyo), and Tokyo Endpoints (Tuners)].

**Example 2. Basic Strategy for Using Proxies**

In this situation, using proxies means that files are transferred overseas only when they are requested. If you used a mirror instead, files would be transferred according to regularly scheduled times, even if no endpoints had requested an update. Proxies require less management than repeaters and mirrors, and if one or more transmitters were added to the central office in London, no special configuration would need to be done in Tokyo.
Example 3. You want to make channels available to endpoints outside your firewall, but you don’t want to allow direct, unmonitored access to your internal master, which might contain sensitive information, such as a database of credit card numbers.

Example 3. Reverse Proxy Outside a Firewall

In this situation, the reverse proxy acts much like a mirror. You give endpoints the URL of the proxy rather than the master, and the endpoints never realize that the proxy forwards requests to a (possibly secure) master. If necessary for load balancing, you could use multiple reverse proxies with a third-party redirector (to accomplish DNS round-robin, for example). You can use an SSL connection between the master and the reverse proxy or between the reverse proxy and the endpoints, or both. You can also preload channels on the reverse proxy, so that even the first time a channel is requested, the download is quick.
Example 4. You want to put a transmitter in one of your customers’ sites. You want all downloads, including the first one, to be as quick as possible, and you want the customers’ endpoints to point to the transmitter at the customer site.

Example 4. Mirror at One of Your Customers’ Sites

In this situation, only one mirror uses overseas bandwidth to replicate channels from the master. Additional transmitters provide load balancing by replicating channels from the Tokyo mirror. You could alternatively use proxies instead of or in addition to the load-balancing mirrors or repeaters.

Note regarding security settings: Subscribe permissions work differently for repeaters than for mirrors. With repeaters, the endpoint sends the subscribe password to the master. No permissions check is then made on the repeaters. With mirrors, subscribe permissions are replicated on each mirror, since endpoints contact the mirror directly. With regard to SSL connections, you can use SSL connections with both repeaters and mirrors.
Determining Whether Your System Can Support the Logging Feature

Overview of the issues involved in configuring log collection on endpoints. If you use the logging feature that is part of the Inventory module, endpoints will collect log entries in a file and then send the file to a Logging plug-in on the master transmitter. The plug-in then inserts the log entries into a database. But the database can only insert a limited number of entries per second. If too many log files are sent to the plug-in at once, so that they cannot all be immediately inserted into the database, then some files are placed in the transmitter’s disk-based queue, until they can be inserted in the database. To keep the system running smoothly, you need to keep the queue from growing too large, and you need to keep the database from getting too full.

Issue 1: Determining the database insertion rate. The first step to managing your system is to find out how many entries your database can insert per second. If your database cannot insert entries quickly enough, the log files in the queue will build up over time. If the number of files in the queue or the amount of disk space used by the queue grows too large, your transmitter could run out of memory.

To give you an idea of the insertion rate: A Windows machine running Microsoft SQL Server 2000, with dual-processors, a processor speed of 1 GHz, and 2 GB of RAM with an ultra-SCSI 10K RPM Raid level 0 array can insert 2,200 log entries per second. This insertion rate assumes that no other plug-ins, such as the Inventory plug-in, are inserting records into the database at the same time.

For assistance with figuring out how many entries your database can insert per second, contact Marimba’s Customer Support.

Issue 2: Determining the volume of log entries generated. Once you find out what your database’s insertion rate is, you need to find out whether that insertion rate will be sufficient to handle the number of log entries your system generates. Several factors come into play. You will need to gather the following information about your endpoints:

- Total number of endpoints that will be generating logs.
- Number of log entries generated by one endpoint during one subscription run (assuming you use Marimba’s Subscription module to install packaged applications). When you configure logging, the setting that has the biggest impact here is the log severity level.

Warning: You need to either use the severity level called major or, if you use the level called audit, then set very specific ranges of log entries to collect. The audit severity level collects a large volume of log entries,
regardless of whether or not applications fail to be installed. Using the AUDIT severity level while not restricting the log ID ranges can cause your database to fill up very quickly.

- Number of retries that the Subscription module performs in the event of a failure to install a packaged application. The default is 5.
- The time interval between retries. This time period has an effect on whether the queue can be processed before the next influx of log entries is created, when the Subscription module runs again. The default is one minute.
- The update interval for Subscription. By default, Subscription runs every 90 minutes on the endpoints.

For assistance, be sure to contact Marimba’s Customer Support, who will help you determine what these numbers are. These numbers will help you determine whether the database can handle the insertion rate and whether a queue will begin to accumulate on the transmitter or transmitters in your system.

**Issue 3: Controlling the queue size on the transmitter.** If you do anticipate that a queue will build up, you will need to gather the following information, to help you determine what configuration settings to use regarding queue size:

- Number of transmitters that will be inserting log entries directly into the database
- Number of endpoints per transmitter that insert logs directly into the database (for repeaters that forward log files to a master, count the endpoints for all the repeaters that forward files to the master)

Again, your Customer Support engineer will help you determine these numbers.

In order to prevent the transmitter from running out of memory, you can use a Logging plug-in configuration setting that will limit the number of files allowed in the queue; the default is 500,000 files. (You can also use a setting to limit the amount of disk space that can be used by the queue; the default is 100 MB.) For each of these limits, once the limit is met, no more log files will be accepted into the queue. (Information about how to use these configuration settings is provided in the Report Center Administrator’s Guide, available on Marimba’s Documentation page (www.marimba.com/doc/).

**Issue 4: Controlling the size of the database table.** The other factor that might need monitoring is how quickly the database grows, in terms of both disk space used and number of records. By default the database table for the logging feature (the mastlog table) can accommodate up to 10 million records, which might take up to 5 GB of disk space.
By default, a delete script is run that creates a regularly scheduled job to delete records from your database on an hourly basis (because the Subscription module runs every 90 minutes, by default). If you want to change the schedule for deleting records, see the Report Center Administrator’s Guide, available on Marimba’s Documentation page (www.marimba.com/doc/).

Keep in mind that if you set the log message severity level to AUDIT and collect log entries for all ranges, the speed at which the database will use up disk space could be 100 times faster than if you set the severity level to MAJOR. Therefore, if you need to collect log entries at the AUDIT level, be sure to specify very limited ranges of log entries. For example, if you set ranges so that only the channel state information is collected, then no matter how many files in a packaged application fail to be installed, you will get only a single installation failure. If you set the range to collect all AUDIT-level log entries, then if there are 1,000 files in the packaged application, you would get 4,000 log entries (that is, 4 log entries for each object, regardless of whether the object failed to install).